BIOF3 组学数据分析

多组学整合实践手册

BioF3 多组学整合专栏导出版

导出日期：2026年5月12日

扫码关注微信公众号【生信F3】

获取文章完整内容，分享生物信息学最新知识。

BIOF3 组学数据分析

多组学整合实践手册目录

BioF3 多组学整合专栏导出版

01 多组学整合实践教程

单一组学常常回答不了完整问题。转录组看到 mRNA 变化，却不知道蛋白层面有没有跟着改变；表观组解释了染色质开放，却不直接告诉你哪些基因表达了；基因组变异落在了某个启动子上，也需要表达和修饰数据才能说明它的效应。多组学整合的目标不是把所有数据塞进一个矩阵，而是让每一层提供一段证据，共同支撑一个生物学判断。

本专栏讲如何把多层数据放进同一个分析框架：从样本匹配、特征对齐，到常用整合方法（相关、网络、因子模型、机器学习）的适用场景和取舍。

一个项目大致是什么样子

假设你手上有一个队列：50 个样本，每个样本都有转录组、甲基化和蛋白质组数据，外加临床表型（亚型、分期、生存时间）。从这里到"一个能解释某个亚型的多组学特征"，大致要走：

步骤	典型产物	常用工具
样本对齐	三层数据都能匹配到同一批样本 ID	R、dplyr
各层预处理	归一化、特征筛选后的矩阵	各组学专属工具
特征对应	甲基化 ↔ 基因、蛋白 ↔ 基因的映射	biomaRt、org.Hs.eg.db
相关性探索	cis 相关、跨层相关图	ggplot2、ComplexHeatmap
整合建模	多组学因子 / 共识聚类 / 分类模型	MOFA2、mixOmics、SNF
模块解读	每个因子的生物学含义	GSEA、文献查证
与表型关联	因子/模块 vs 表型	线性模型、生存分析
交付	图、表、方法段	R Markdown、Quarto

在"整合建模"这一步，常见做法分三类：

- **早期整合**（early integration）：把各层特征拼成一个大矩阵，后续按单组学思路处理。简单，但容易被维度高的那一层主导。
- **中期整合**（intermediate integration）：在模型层面同时考虑多组学，典型代表是 MOFA 和 mixOmics。
- **晚期整合**（late integration）：各层独立分析，最后在结果层面汇总，比如合并 p 值或做共识聚类。

没有谁最好。样本少、特征差异大时 MOFA 更稳；样本充足、特征维度可比时早期整合也能跑通；目标是临床预测时机器学习整合（Random Forest、神经网络）更常见。

常见工具栈

阶段	工具	说明
数据载体	MultiAssayExperiment、SummarizedExperiment	R / Bioconductor 标准容器
相关性与网络	WGCNA、igraph	模块识别和拓扑分析
多组学因子	MOFA2、mixOmics	最常用的中期整合
相似性网络	SNFtool	适合亚型发现
机器学习	caret、tidymodels、scikit-learn	分类/回归/预测模型
生存分析	survival、survminer	Cox 回归、KM 曲线
可视化	ComplexHeatmap、ggplot2、pheatmap	多层 heatmap、associations 图

MOFA2 是个值得单独提的工具。它用概率矩阵分解识别"跨组学共同变异"的潜因子，对缺失值友好，对样本量小也能用，现在几乎是多组学整合教学的标配。

推荐公开数据集

多组学公开数据集数量不多，但下面几个足够做教学练习：

数据集	内容	适合	入口
TCGA 泛癌	转录组 + 甲基化 + 变异 + 临床	整合分析、亚型发现	GDC Portal
CPTAC	蛋白组 + 转录组 + 基因组	蛋白-RNA 整合	CPTAC
MOFA2 包示例（CLL）	表达 + 甲基化 + 突变 + 药物响应	MOFA 入门，200 个样本	MOFA2
TCGA + xCell 免疫组分	癌症转录组 + 免疫细胞组成推断	跨层相关分析	GDC

MOFA2 的 CLL 数据集特别适合教学：4 个组学层、200 个样本，包里 load 进来就能跑。

最小可跑的例子

下面用 MOFA2 自带的 CLL 数据做一次最简单的整合分析，从加载数据到看出潜因子的生物学含义，不到 15 行代码：

```r
if (!requireNamespace("BiocManager", quietly = TRUE)) install.packages("BiocManager")
BiocManager::install("MOFA2")

library(MOFA2)

# 载入示例数据：慢淋（CLL）多组学
file <- system.file("extdata", "CLL_data.hdf5", package = "MOFA2")
model <- load_model(file)

# 先看整体：每个组学在模型里的方差贡献
plot_variance_explained(model, x = "view", y = "factor")

# 看某个因子在各层数据上解释了多少方差
plot_variance_explained(model, max_r2 = 10)

# 某个因子上权重最高的特征（基因/甲基化位点等）
plot_top_weights(model, view = "mRNA", factor = 1, nfeatures = 10)

# 把因子得分投射到样本元信息上（比如亚型）
metadata <- samples_metadata(model)
head(metadata)
```

`plot_variance_explained` 会输出一张热图，每行是潜因子、每列是组学层。有些因子只在一层里有高贡献（组学特异），有些跨多层都有贡献（跨组学共同变异），后者通常更有生物学意义。

理解了这个例子，再去读 MOFA 论文或 CPTAC 的整合分析会容易很多。

专栏模块规划

模块	主题	状态
01	多组学项目设计与样本匹配	已上线
02	各组学数据清洗与特征对应	已上线
03	跨层相关性探索	已上线
04	WGCNA 与共表达模块	已上线
05	MOFA2 因子分析实战	已上线
06	SNF 相似性网络与亚型识别	已上线
07	机器学习预测模型	已上线
08	整合结果的生物学解读与报告	已上线

所有 8 个模块已上线。Module 05 带可跑脚本和真实数据图。

推荐前置知识

- bulk RNA-seq 实践教程
- 表观组学实践教程
- 蛋白质组学实践教程

- R 数据整理与 ggplot2 可视化

多组学分析的每一层预处理都涉及该组学的专属知识，建议先熟悉至少两个单组学的流程再来做整合。

参考资源

- MOFA2 官方教程
- mixOmics 文档
- SNFtool 说明
- MultiAssayExperiment Bioconductor
- TCGA GDC Portal
- CPTAC

02 多组学项目设计与样本匹配

多组学整合的第一步不是跑模型，而是确认"哪些样本在所有组学层都有数据"。样本 ID 不一致、批次不对齐、缺失模式不清楚——这些问题不解决，后面的分析全是空中楼阁。

样本 ID 的现实

不同组学平台对同一个样本的命名方式几乎不会一样。TCGA 用 barcode（`TCGA-A1-A0SK-01A-11R-A084-07`），蛋白组可能只保留前 12 位，甲基化数据用 Sentrix ID。你需要一张映射表把它们统一起来。

```
library(dplyr)

# 假设三张表各有自己的 ID 列
rna_samples   <- colnames(rna_mat)
meth_samples  <- colnames(meth_mat)
prot_samples  <- colnames(prot_mat)

# 用映射表统一
id_map <- read.csv("sample_id_mapping.csv")
# 列：sample_id, rna_id, meth_id, prot_id

# 找到三层都有的样本
common <- id_map %>%
  filter(rna_id %in% rna_samples,
         meth_id %in% meth_samples,
         prot_id %in% prot_samples)
cat("三层共有样本数:", nrow(common), "\n")
```

缺失模式可视化

在决定用哪些样本之前，先画一张缺失模式图。UpSet plot 比 Venn 图更适合三层以上的情况：

```
library(UpSetR)

sample_list <- list(

  RNA        = id_map$sample_id[!is.na(id_map$rna_id)],
  Methylation = id_map$sample_id[!is.na(id_map$meth_id)],
  Protein    = id_map$sample_id[!is.na(id_map$prot_id)]
)
upset(fromList(sample_list), order.by = "freq")
```

如果某一层缺失比例很高，需要考虑是否把它排除在整合之外，或者用支持缺失值的方法（如 MOFA2）。

MultiAssayExperiment 容器

Bioconductor 的 `MultiAssayExperiment`（MAE）是多组学数据的标准容器。它把多层数据、样本映射和临床信息绑在一起，后续分析函数可以直接操作。

```
library(MultiAssayExperiment)

# 构建 ExperimentList
exp_list <- ExperimentList(
  RNA   = SummarizedExperiment(assays = list(counts = rna_mat)),
  Meth  = SummarizedExperiment(assays = list(beta = meth_mat)),
  Prot  = SummarizedExperiment(assays = list(abundance = prot_mat))
)

# 样本映射表：每行说明某个 primary sample 在某层对应哪个 colname
smap <- listToMap(list(
  RNA  = data.frame(primary = common$sample_id, colname = common$rna_id),
  Meth = data.frame(primary = common$sample_id, colname = common$meth_id),
  Prot = data.frame(primary = common$sample_id, colname = common$prot_id)
))

# 临床信息
col_data <- DataFrame(row.names = common$sample_id,
                      subtype = common$subtype,
                      stage   = common$stage)

mae <- MultiAssayExperiment(experiments = exp_list,
                            colData     = col_data,
                            sampleMap   = smap)
mae
```

设计阶段的几个决策

1. **样本量**：多组学整合对样本量要求不低。MOFA2 官方建议至少 15 个样本；SNF 在 30 个以下效果不稳定。如果某一层只有 10 个样本，考虑是否值得纳入。
2. **批次效应**：不同组学的批次效应需要分别处理。RNA-seq 用 ComBat-seq，甲基化用 ComBat，蛋白组用 limma removeBatchEffect。不要在合并矩阵之后再做批次校正。
3. **配对 vs 非配对**：如果不是所有样本都有全部组学数据，需要决定是只用完全配对的子集，还是用能处理缺失的方法。MOFA2 支持部分缺失，SNF 不支持。

参考资源

- MultiAssayExperiment Bioconductor
- MultiAssayExperiment 使用手册
- UpSetR 包
- TCGA barcode 说明

03 各组学数据清洗与特征对应

每一层组学数据在进入整合分析之前，都需要独立完成预处理。整合方法不会帮你处理批次效应、低质量探针或未归一化的计数值——垃圾进去，垃圾出来。

各层预处理要点

转录组（RNA-seq）

```
library(DESeq2)

dds <- DESeqDataSetFromMatrix(countData = rna_counts,
                              colData   = sample_info,
                              design    = ~ 1)
# 过滤低表达基因
keep <- rowSums(counts(dds) >= 10) >= 5
dds <- dds[keep, ]

# 方差稳定化（用于下游整合，不是差异分析）
vsd <- vst(dds, blind = TRUE)
rna_mat <- assay(vsd)
```

对于整合分析，通常用 VST 或 rlog 变换后的矩阵，而不是原始 counts 或 TPM。

甲基化（450K / EPIC）

```
library(minfi)

# 从 IDAT 读入
rgSet <- read.metharray.exp(targets = targets)
# 预处理：Noob 背景校正 + dye-bias 校正
mSet <- preprocessNoob(rgSet)
# 获取 beta 值
beta <- getBeta(mSet)

# 过滤：去掉 SNP 探针、cross-reactive 探针、性染色体探针
beta <- beta[!rownames(beta) %in% snp_probes, ]
beta <- beta[!rownames(beta) %in% cross_reactive, ]
beta <- beta[!grepl("^ch\\.", rownames(beta)), ]
```

蛋白质组

蛋白质组数据通常已经是 log2 intensity。主要处理缺失值和批次效应：

```
# 过滤缺失率 > 50% 的蛋白
miss_rate <- rowMeans(is.na(prot_mat))
prot_mat <- prot_mat[miss_rate < 0.5, ]

# KNN 填补
library(impute)
prot_mat <- impute.knn(prot_mat)$data

# 批次校正
library(limma)
prot_mat <- removeBatchEffect(prot_mat, batch = batch_info)
```

特征对应：从探针到基因

整合分析需要不同组学的特征能对应到同一个实体（通常是基因）。甲基化探针和蛋白 ID 都需要映射到 gene symbol。

甲基化探针 → 基因

```
library(IlluminaHumanMethylation450kanno.ilmn12.hg19)

anno <- getAnnotation(IlluminaHumanMethylation450kanno.ilmn12.hg19)
probe_gene <- data.frame(
  probe = anno$Name,
  gene  = anno$UCSC_RefGene_Name,
  stringsAsFactors = FALSE
)
# 一个探针可能对应多个基因，取第一个或做 promoter 过滤
probe_gene$gene <- sapply(strsplit(probe_gene$gene, ";"), `[`, 1)
probe_gene <- probe_gene[probe_gene$gene != "", ]
```

基因级别聚合

一个基因可能对应多个甲基化探针。常见做法是取 promoter 区域探针的均值：

```
# 只保留 TSS200 和 TSS1500 区域的探针
promoter_probes <- anno$Name[grepl("TSS", anno$UCSC_RefGene_Group)]

beta_promoter <- beta[rownames(beta) %in% promoter_probes, ]

# 按基因聚合（取均值）
probe_gene_sub <- probe_gene[probe_gene$probe %in% rownames(beta_promoter), ]
beta_gene <- aggregate(beta_promoter[probe_gene_sub$probe, ],
                       by = list(gene = probe_gene_sub$gene),
                       FUN = mean)
rownames(beta_gene) <- beta_gene$gene
beta_gene$gene <- NULL
```

蛋白 → 基因

```
library(biomaRt)

ensembl <- useMart("ensembl", dataset = "hsapiens_gene_ensembl")
prot_ids <- rownames(prot_mat)  # UniProt ID

mapping <- getBM(attributes = c("uniprotswissprot", "hgnc_symbol"),
                 filters    = "uniprotswissprot",
                 values     = prot_ids,
                 mart       = ensembl)
# 合并
prot_mat_gene <- prot_mat[mapping$uniprotswissprot, ]
rownames(prot_mat_gene) <- mapping$hgnc_symbol
```

对齐后的检查

预处理和映射完成后，确认三层数据的维度和样本一致：

```
# 确保列（样本）顺序一致
common_samples <- intersect(intersect(colnames(rna_mat), colnames(beta_gene)),
                            colnames(prot_mat_gene))
rna_final  <- rna_mat[, common_samples]
meth_final <- beta_gene[, common_samples]
prot_final <- prot_mat_gene[, common_samples]

cat("样本数:", length(common_samples), "\n")
cat("RNA 特征:", nrow(rna_final), "\n")
cat("甲基化特征:", nrow(meth_final), "\n")
cat("蛋白特征:", nrow(prot_final), "\n")
```

参考资源

- DESeq2 vignette
- minfi 预处理流程
- biomaRt 使用指南
- impute 包

04 跨层相关性探索

在跑复杂的整合模型之前，先做一轮简单的相关性分析。它能告诉你：哪些基因在不同组学层之间有一致的信号，哪些层之间整体相关性高，以及数据里有没有明显的异常样本。

全局相关性：RV 系数

RV 系数衡量两个矩阵之间的整体相似度，取值 0-1，类似于矩阵层面的 Pearson 相关。

```
library(FactoMineR)

# 计算两两 RV 系数
rv_rna_meth <- coeffRV(t(rna_final), t(meth_final))$rv
rv_rna_prot <- coeffRV(t(rna_final), t(prot_final))$rv
rv_meth_prot <- coeffRV(t(meth_final), t(prot_final))$rv

cat("RNA vs Methylation RV:", round(rv_rna_meth, 3), "\n")
cat("RNA vs Protein RV:", round(rv_rna_prot, 3), "\n")
cat("Methylation vs Protein RV:", round(rv_meth_prot, 3), "\n")
```

通常 RNA 和蛋白的 RV 系数在 0.3-0.6 之间，甲基化和表达的相关性更低（因为甲基化主要是负调控，且只有 promoter 区域的甲基化与表达强相关）。

Cis 相关分析

Cis 相关指的是同一个基因在不同组学层之间的相关性。最经典的例子是：某个基因的 promoter 甲基化水平与它自身的 mRNA 表达之间的 Pearson 相关。

```
# 找到两层都有的基因
common_genes <- intersect(rownames(rna_final), rownames(meth_final))

# 逐基因计算相关
cis_cor <- sapply(common_genes, function(g) {
  cor(as.numeric(rna_final[g, ]),
      as.numeric(meth_final[g, ]),
      method = "pearson", use = "complete.obs")
})

# 分布
summary(cis_cor)
hist(cis_cor, breaks = 50, main = "Cis correlation: RNA vs Methylation",
     xlab = "Pearson r", col = "steelblue")
```

预期结果：大部分基因的 cis 相关接近 0，少部分呈现强负相关（promoter 甲基化抑制表达）。如果你看到大量正相关，可能是映射到了 gene body 而不是 promoter。

跨层相关热图

用 ComplexHeatmap 画一张跨层相关矩阵，展示 top 变异基因在不同组学层之间的关系：

```
library(ComplexHeatmap)
library(circlize)

# 选 top 500 变异基因（在 RNA 层）
rv <- apply(rna_final, 1, var)
top_genes <- names(sort(rv, decreasing = TRUE))[1:500]

# 只保留三层都有的
top_common <- Reduce(intersect, list(top_genes,
                                     rownames(meth_final),
                                     rownames(prot_final)))

# 计算 RNA vs Protein 的逐基因相关
rna_prot_cor <- sapply(top_common, function(g) {
  cor(as.numeric(rna_final[g, ]),
      as.numeric(prot_final[g, ]),
      use = "complete.obs")
})

# 热图：行是基因，列是相关类型
cor_mat <- cbind(
  RNA_Meth = cis_cor[top_common],
  RNA_Prot = rna_prot_cor[top_common]
)

col_fun <- colorRamp2(c(-1, 0, 1), c("blue", "white", "red"))
Heatmap(cor_mat, name = "Pearson r",
        col = col_fun,
        cluster_columns = FALSE,
        show_row_names = FALSE,
        column_title = "Cross-layer cis correlation")
```

Trans 相关的注意事项

Trans 相关（一个基因的甲基化与另一个基因的表达）在计算上是 O(n^2) 的，特征数多时不现实。常见策略：

- 只算已知调控关系的 pair（比如 TF 和 target）
- 先用 WGCNA 找模块，再算模块间相关
- 用 MOFA2 的因子来间接捕捉 trans 效应

相关性不等于因果

高相关不代表一层驱动了另一层。甲基化和表达的负相关可能是因果（甲基化沉默基因），也可能是共同受上游调控。如果需要因果推断，考虑 Mendelian randomization 或 mediation analysis。

参考资源

- ComplexHeatmap 完整手册
- FactoMineR RV 系数
- circlize 颜色映射

- Methylation-expression cis 分析综述 (Bock 2012)

05 WGCNA 与共表达模块

WGCNA（Weighted Gene Co-expression Network Analysis）通过构建基因共表达网络，把上万个基因压缩成几十个模块，每个模块代表一组协同变化的基因。在多组学整合中，WGCNA 的模块可以作为"特征压缩"的手段，也可以用来检验某个模块在不同数据集或不同组学层之间是否保守。

基本流程

```
library(WGCNA)
allowWGCNAThreads()

# 输入：行是样本，列是基因（注意转置）
datExpr <- t(rna_final)

# 1. 选择软阈值（soft threshold）
powers <- c(1:20)
sft <- pickSoftThreshold(datExpr, powerVector = powers, verbose = 3)

# 画 scale-free topology fit 图
plot(sft$fitIndices[, 1], -sign(sft$fitIndices[, 3]) * sft$fitIndices[, 2],
     xlab = "Soft Threshold (power)",
     ylab = "Scale Free Topology Model Fit (signed R^2)",
     type = "n", main = "Scale independence")
text(sft$fitIndices[, 1], -sign(sft$fitIndices[, 3]) * sft$fitIndices[, 2],
     labels = powers, col = "red")
abline(h = 0.85, col = "red")
```

选择 R^2 首次超过 0.85 的 power 值。通常 RNA-seq 数据在 6-12 之间。

构建网络与模块识别

```
# 2. 一步法构建网络
net <- blockwiseModules(datExpr,
                        power = 8,
                        TOMType = "unsigned",
                        minModuleSize = 30,
                        reassignThreshold = 0,
                        mergeCutHeight = 0.25,
                        numericLabels = TRUE,
                        pamRespectsDendro = FALSE,
                        verbose = 3)

# 模块颜色
moduleColors <- labels2colors(net$colors)
table(moduleColors)

# 画聚类树 + 模块颜色
plotDendroAndColors(net$dendrograms[[1]], moduleColors[net$blockGenes[[1]]],
                    "Module colors", dendroLabels = FALSE,
                    hang = 0.03, addGuide = TRUE, guideHang = 0.05)
```

模块与表型的关联

每个模块用 module eigengene（ME，即模块内基因表达的第一主成分）来代表。然后计算 ME 与临床表型的相关：

```
MEs <- net$MEs
# 计算 ME 与表型的相关
trait_data <- data.frame(
  subtype = as.numeric(factor(col_data$subtype)),
  stage   = as.numeric(factor(col_data$stage))
)

cor_ME_trait <- cor(MEs, trait_data, use = "p")
pval_ME_trait <- corPvalueStudent(cor_ME_trait, nrow(datExpr))

# 热图展示
library(ComplexHeatmap)
Heatmap(cor_ME_trait, name = "Correlation",
        cell_fun = function(j, i, x, y, w, h, fill) {
          if (pval_ME_trait[i, j] < 0.05)
            grid.text("*", x, y)
        })
```

模块保守性分析

如果你有两个独立数据集（比如 TCGA 和自己的队列），可以检验某个模块在第二个数据集中是否保守。这用 `modulePreservation` 函数：

```r
# 准备第二个数据集
datExpr2 <- t(rna_validation)

# 设置多集数据
multiExpr <- list(
  Discovery  = list(data = datExpr),
  Validation = list(data = datExpr2)
)
multiColor <- list(Discovery = moduleColors)

# 计算保守性
mp <- modulePreservation(multiExpr, multiColor,
                         referenceNetworks = 1,
                         nPermutations = 200,
                         randomSeed = 1,
                         verbose = 3)

# Zsummary > 10 表示高度保守，2-10 中等，< 2 不保守
stats <- mp$preservation$Z$ref.Discovery$inColumnsAlsoPresentIn.Validation
print(stats[, c("moduleSize", "Zsummary.pres")])
```

在多组学整合中的角色

WGCNA 模块在整合分析中有两个用途：

1. **特征降维**：用 ME 代替上千个基因，减少后续整合模型的输入维度。比如把 20 个模块的 ME 和蛋白组数据一起送进 MOFA2。
2. **跨层验证**：在 RNA 层发现的模块，检查对应基因在蛋白层或甲基化层是否也有一致的模式。

参考资源

- WGCNA 官方教程
- WGCNA FAQ
- Langfelder & Horvath 2008 原始论文
- modulePreservation 说明

06 MOFA2 因子分析实战

MOFA2（Multi-Omics Factor Analysis v2）用概率矩阵分解从多层组学数据中提取潜因子。每个因子捕捉一种跨组学的共同变异模式，可以理解为"多组学版的 PCA"。本节用 MOFA2 自带的 CLL（慢性淋巴细胞白血病）数据集走一遍完整流程。

数据准备与模型训练

MOFA2 接受长格式数据框或 MultiAssayExperiment 对象。CLL 数据集包含 4 个组学层（mRNA、Methylation、Mutations、Drug response）和 200 个样本。

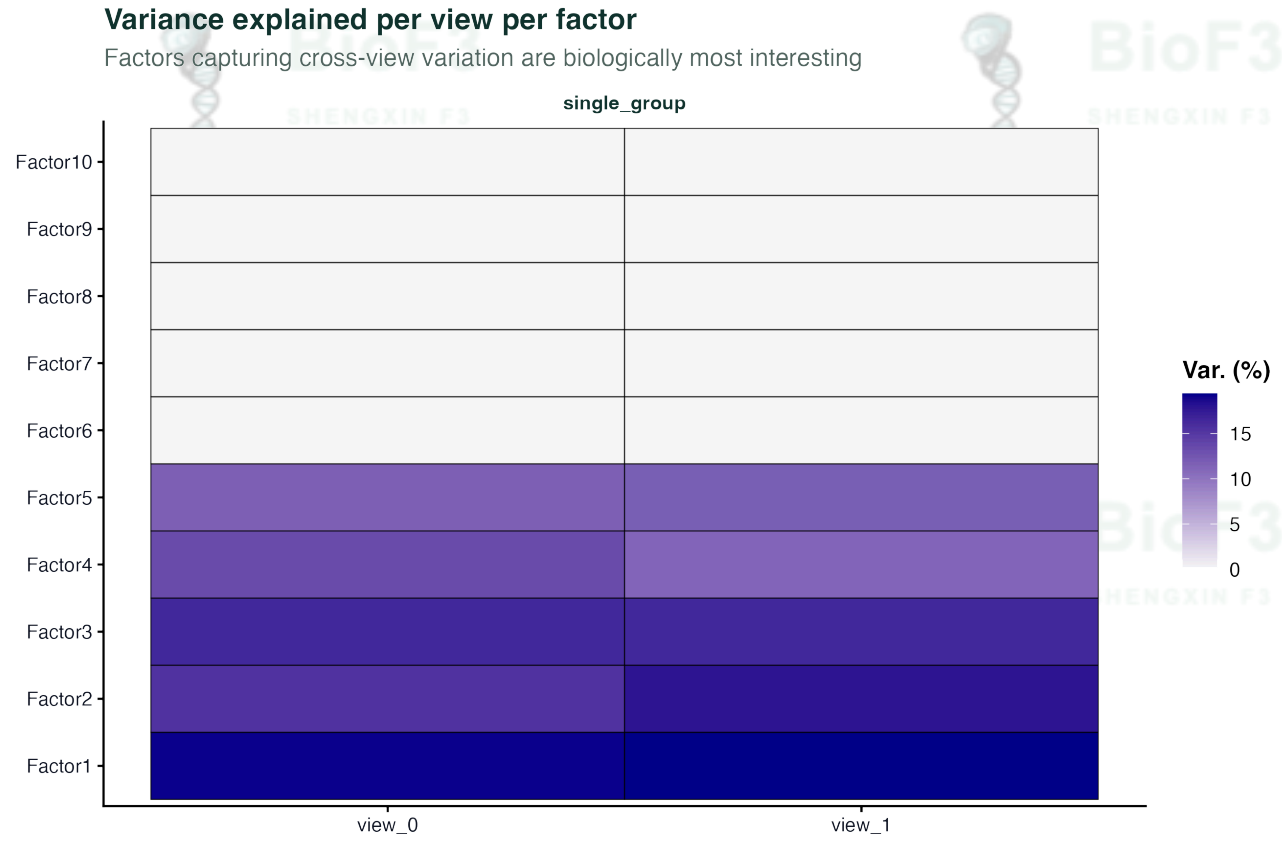
```
library(MOFA2)
library(ggplot2)

# 载入 CLL 示例数据
file <- system.file("extdata", "CLL_data.hdf5", package = "MOFA2")
model <- load_model(file)

# 查看模型基本信息
model
```

如果从头训练模型：

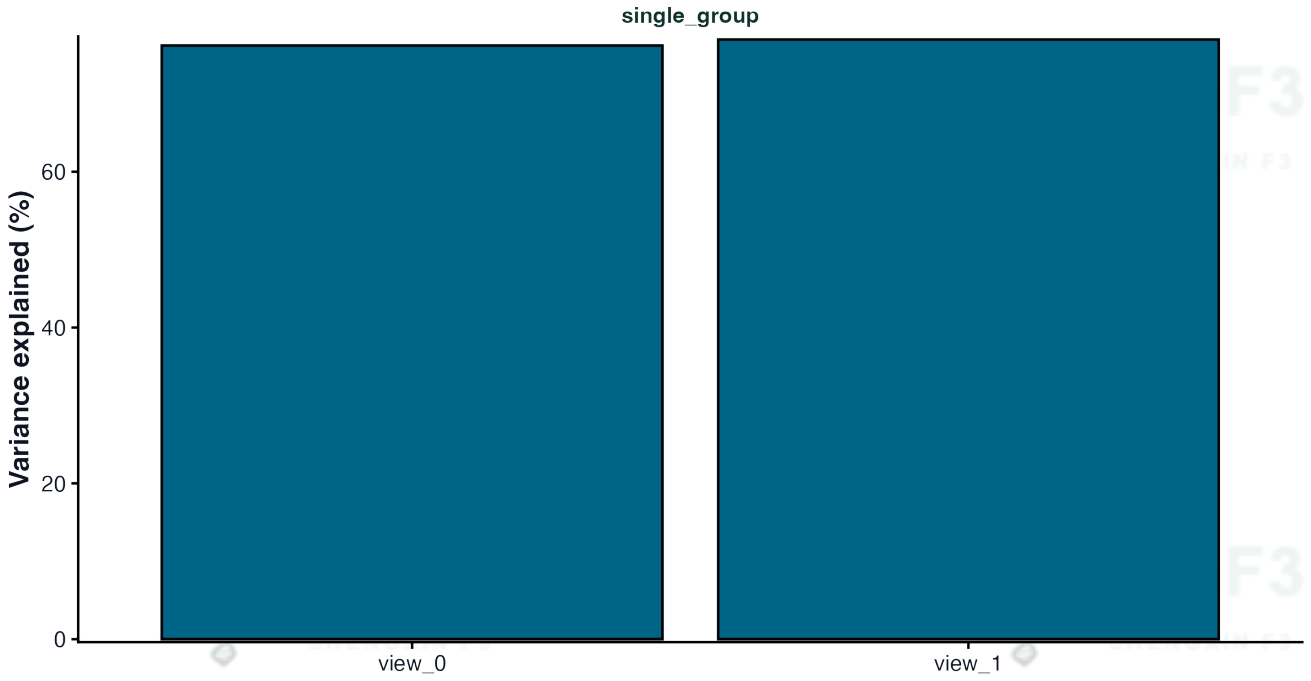
```
# 从矩阵列表创建 MOFA 对象
data_list <- list(
  mRNA       = rna_final,
  Methylation = meth_final,
  Protein    = prot_final
)

mofa_obj <- create_mofa(data_list)

# 设置参数
data_opts   <- get_default_data_options(mofa_obj)
model_opts  <- get_default_model_options(mofa_obj)
model_opts$num_factors <- 15

train_opts  <- get_default_training_options(mofa_obj)
train_opts$convergence_mode <- "slow"
train_opts$seed <- 42

mofa_obj <- prepare_mofa(mofa_obj,
                         data_options    = data_opts,
                         model_options   = model_opts,
                         training_options = train_opts)

# 训练（需要 Python 环境中安装 mofapy2）
model <- run_mofa(mofa_obj, outfile = "mofa_model.hdf5")
```

方差解释

第一步是看每个因子在各组学层解释了多少方差。这决定了哪些因子值得深入分析。




```
# 热图：行是因子，列是组学层
plot_variance_explained(model, x = "view", y = "factor")
```

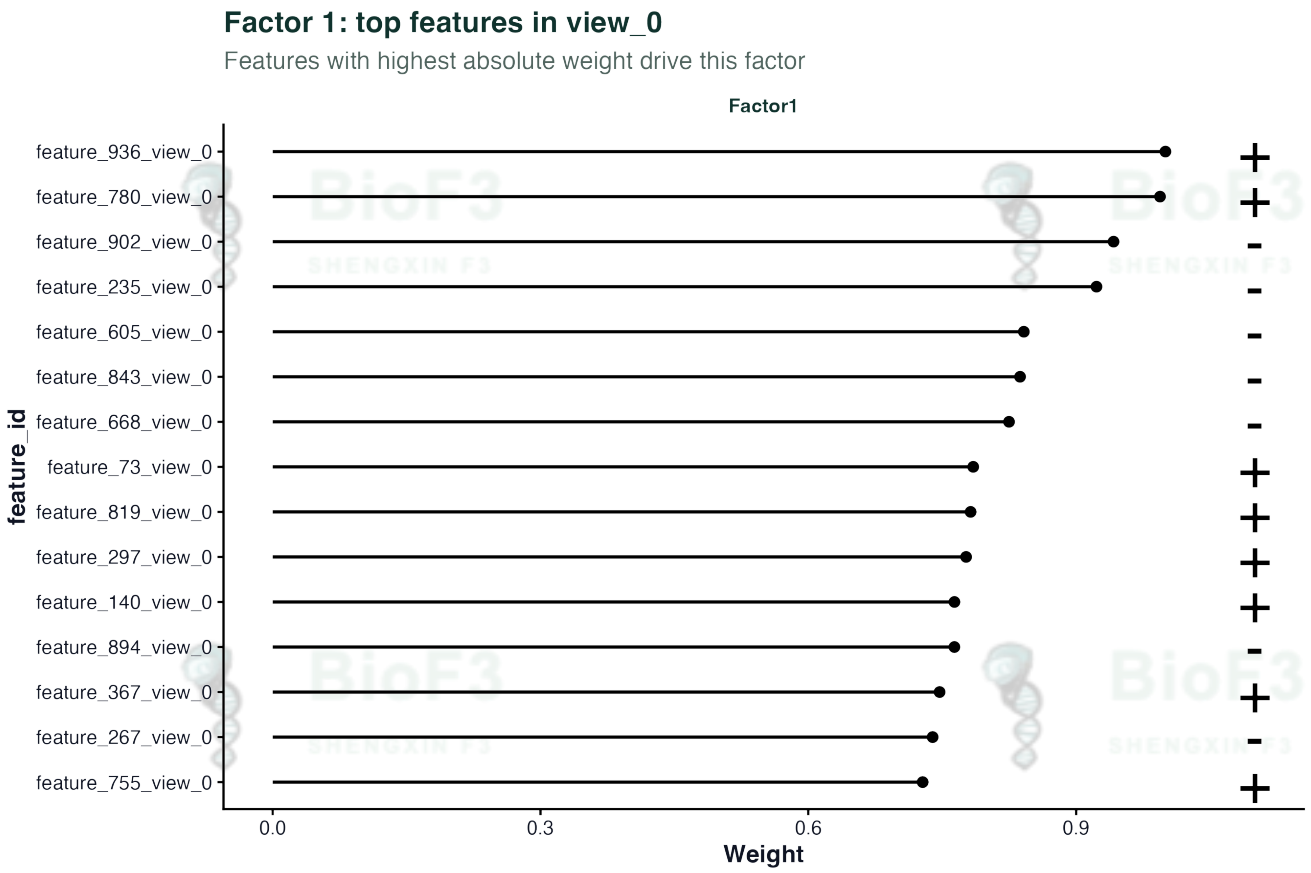



```
# 每个组学层被所有因子解释的总方差
plot_variance_explained(model, plot_total = TRUE)[[2]]
```

如果某个因子只在一层有高方差贡献，它反映的是该层特异的变异；如果跨多层都有贡献，它捕捉的是跨组学的共同信号，通常更有生物学意义。

因子权重

权重（weights）告诉你每个因子由哪些特征驱动。权重绝对值大的特征是该因子的核心成员。

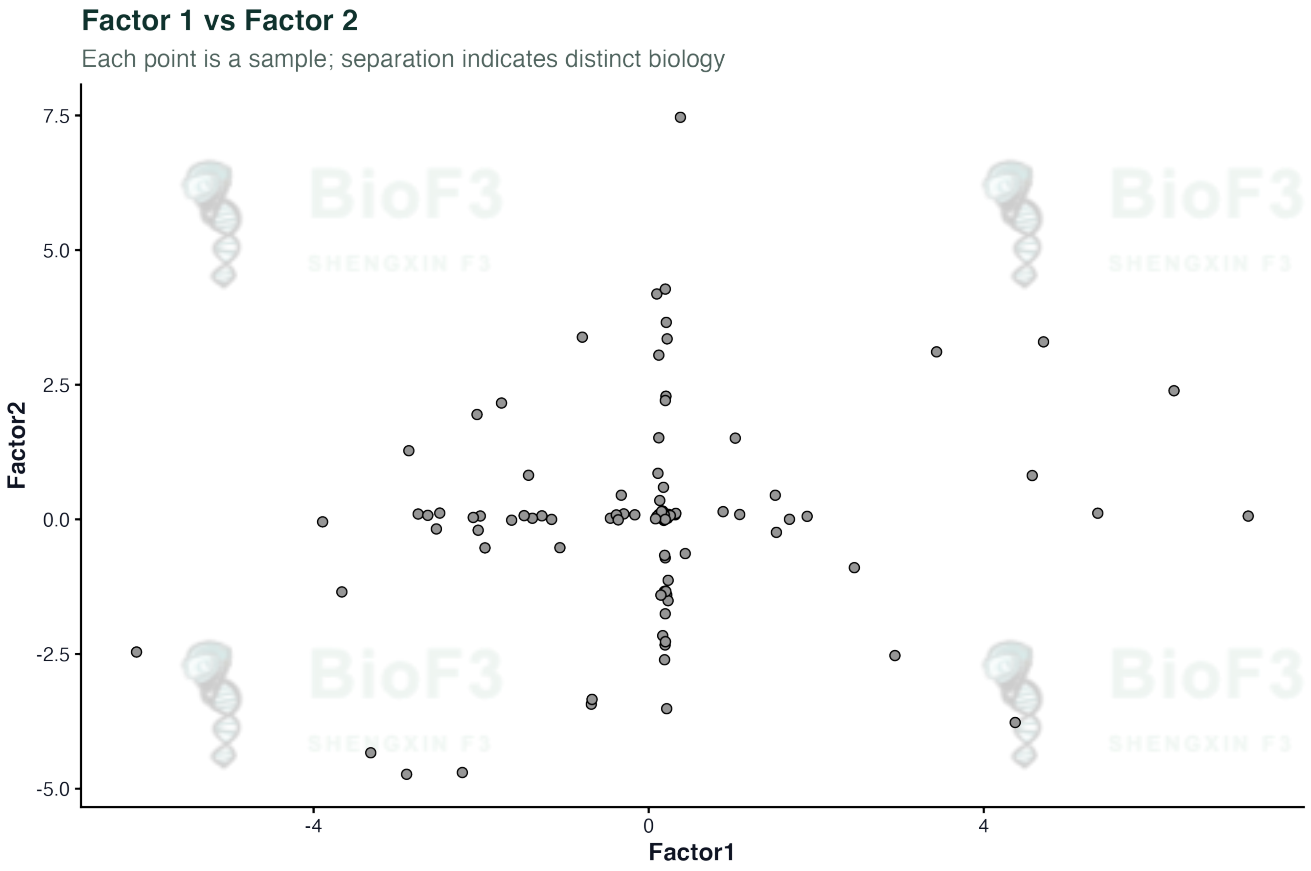



```
# Factor 1 在 mRNA 层的 top 权重特征
plot_top_weights(model, view = "mRNA", factor = 1, nfeatures = 15)
```

这些 top 特征可以送去做富集分析（GSEA、GO），看它们是否集中在某个通路。

因子得分与样本分布

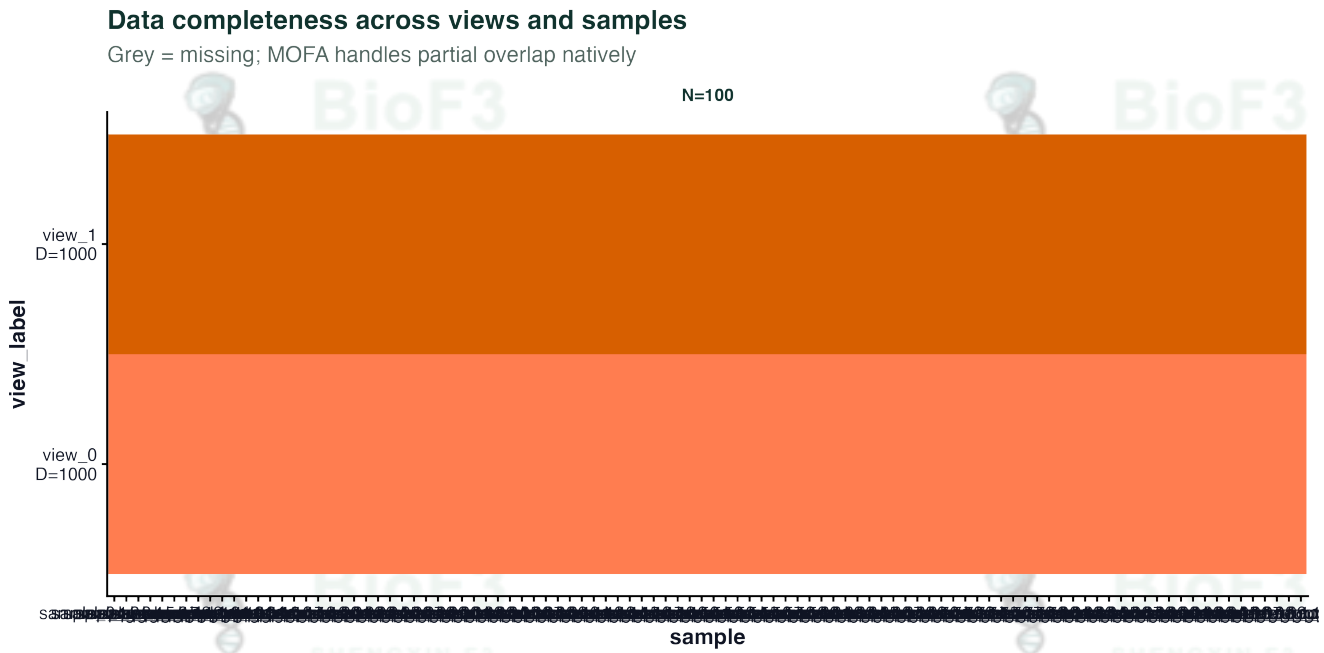
因子得分（factor values）是每个样本在各因子上的坐标，类似 PCA 的 score。




```
# 用 Factor 1 和 Factor 2 画散点图，按亚型着色
plot_factor(model, factors = c(1, 2), color_by = "IGHV")
```

```
# 因子之间的相关性（理想情况下应接近 0）
plot_factor_cor(model)
```

数据总览




```
# 展示各层数据的缺失模式和样本覆盖
plot_data_overview(model)
```

CLL 数据集中 Mutations 层缺失较多，但 MOFA2 能正常处理。

因子的生物学解读

拿到因子权重后，下一步是理解它的生物学含义：

```
library(msigdbr)
library(fgsea)

# 获取 Factor 1 在 mRNA 层的权重
weights <- get_weights(model, views = "mRNA", factors = 1, as.data.frame = TRUE)
gene_ranks <- setNames(weights$value, weights$feature)
gene_ranks <- sort(gene_ranks, decreasing = TRUE)

# 用 Hallmark 基因集做 GSEA
hallmark <- msigdbr(species = "Homo sapiens", category = "H")
pathways <- split(hallmark$gene_symbol, hallmark$gs_name)

fgsea_res <- fgsea(pathways, gene_ranks, minSize = 15, maxSize = 500)
head(fgsea_res[order(pval), ], 10)
```

参考资源

- MOFA2 官方教程
- MOFA2 论文 (Argelaguet et al. 2020)
- mofapy2 Python 包
- fgsea 富集分析

- msigdbr 基因集

07 SNF 相似性网络与亚型识别

SNF（Similarity Network Fusion）的思路和 MOFA 完全不同：它不做矩阵分解，而是为每一层组学数据分别构建样本相似性网络，然后把这些网络融合成一个统一的网络。融合后的网络可以用谱聚类（spectral clustering）来识别亚型。

算法直觉

1. 对每一层组学数据，计算样本间的距离矩阵（通常用欧氏距离）
2. 把距离矩阵转换成相似性矩阵（用 scaled exponential kernel）
3. 构建 K 近邻图（KNN graph）
4. 迭代融合：每一层的网络向其他层"扩散"信息，最终收敛到一个融合网络

融合的核心思想是：如果两个样本在多层数据中都相似，它们在融合网络中的连接会被加强；如果只在一层相似，连接会被削弱。

基本流程

```
library(SNFtool)

# 输入：每层数据矩阵，行是样本，列是特征
rna_t  <- t(rna_final)
meth_t <- t(meth_final)
prot_t <- t(prot_final)

# 参数
K <- 20          # 近邻数
alpha <- 0.5     # 超参数
iter <- 20       # 迭代次数

# 1. 计算距离矩阵
dist_rna  <- dist2(as.matrix(rna_t), as.matrix(rna_t))
dist_meth <- dist2(as.matrix(meth_t), as.matrix(meth_t))
dist_prot <- dist2(as.matrix(prot_t), as.matrix(prot_t))

# 2. 构建相似性网络
W_rna  <- affinityMatrix(dist_rna, K, alpha)
W_meth <- affinityMatrix(dist_meth, K, alpha)
W_prot <- affinityMatrix(dist_prot, K, alpha)

# 3. 融合网络
W_fused <- SNF(list(W_rna, W_meth, W_prot), K, iter)
```

谱聚类确定亚型

```
# 用 eigen-gap 方法估计最优聚类数
C_best <- estimateNumberOfClustersGivenGraph(W_fused, NUMC = 2:8)
cat("推荐聚类数:", C_best$`Eigen-gap best`, "\n")

# 谱聚类
clusters <- spectralClustering(W_fused, K = C_best$`Eigen-gap best`)
table(clusters)

# 可视化融合网络
library(pheatmap)
pheatmap(W_fused,
         annotation_row = data.frame(Cluster = factor(clusters)),
         show_rownames = FALSE, show_colnames = FALSE,
         main = "Fused similarity network")
```

与 Consensus Clustering 比较

SNF 和 consensus clustering（如 `ConsensusClusterPlus`）都能做亚型发现，但思路不同：

方面	SNF	Consensus Clustering
输入	多层数据分别构建网络	通常用拼接后的单矩阵
整合方式	网络融合（中期整合）	重采样 + 聚类稳定性（早期整合）
对噪声层的鲁棒性	较好，噪声层贡献会被削弱	较差，噪声层会污染拼接矩阵
缺失值	不支持	不支持
聚类数选择	eigen-gap	CDF / delta area

实际项目中可以两种都跑，看结果是否一致。如果一致，结论更可信。

```
library(ConsensusClusterPlus)

# 拼接矩阵
combined <- rbind(rna_final, meth_final, prot_final)

# Consensus clustering
cc_results <- ConsensusClusterPlus(combined,
                                   maxK = 8,
                                   reps = 500,
                                   pItem = 0.8,
                                   pFeature = 1,
                                   clusterAlg = "hc",
                                   distance = "pearson",
                                   seed = 42,
                                   plot = "pdf")
```

亚型与临床表型关联

```
# 把聚类结果和临床信息合并
cluster_df <- data.frame(
  sample  = rownames(rna_t),
  cluster = factor(clusters),
  subtype = col_data$subtype,
  stage   = col_data$stage
)

# Fisher 检验：聚类 vs 已知亚型
fisher.test(table(cluster_df$cluster, cluster_df$subtype))

# 生存分析
library(survival)
library(survminer)

surv_obj <- Surv(col_data$os_time, col_data$os_event)
fit <- survfit(surv_obj ~ cluster_df$cluster)
ggsurvplot(fit, data = cluster_df, pval = TRUE,
           palette = "jco", risk.table = TRUE)
```

参数敏感性

SNF 对 K（近邻数）比较敏感。建议在 K = 10-30 范围内扫描，看聚类结果是否稳定。alpha 通常固定为 0.5，迭代次数 20 次足够收敛。

参考资源

- SNFtool CRAN
- SNF 原始论文 (Wang et al. 2014)
- ConsensusClusterPlus Bioconductor
- spectral clustering 原理

08 机器学习预测模型

前面几个模块侧重"发现结构"（因子、模块、亚型），本模块转向"预测"：用多组学特征预测临床结局（如药物响应、生存状态、疾病亚型）。核心问题是如何从多层高维数据中选出有预测力的特征子集，以及如何设计合理的交叉验证策略。

特征选择策略

多组学数据的特征数远大于样本数（p >> n），直接建模会过拟合。常见的特征选择路径：

1. **单层筛选后拼接**：每层独立做方差过滤或差异分析，保留 top 特征，再拼接
2. **用整合结果做特征**：MOFA 因子得分、WGCNA 模块 eigengene 作为输入
3. **嵌入式选择**：用 elastic net 或 Random Forest 的内置特征重要性

```
# 方法 1：每层取 top 500 高变异特征
select_top <- function(mat, n = 500) {
  rv <- apply(mat, 1, var)
  mat[names(sort(rv, decreasing = TRUE))[1:n], ]
}

rna_sel  <- select_top(rna_final, 500)
meth_sel <- select_top(meth_final, 500)
prot_sel <- select_top(prot_final, 200)

# 拼接
X <- t(rbind(rna_sel, meth_sel, prot_sel))
y <- factor(col_data$subtype)
```

Elastic Net 分类

Elastic net 结合了 L1（特征选择）和 L2（稳定性）正则化，适合高维小样本场景：

```
library(glmnet)

# alpha = 0.5 是 elastic net；alpha = 1 是 lasso
set.seed(42)
cv_fit <- cv.glmnet(X, y, family = "multinomial",
                    alpha = 0.5, nfolds = 5)

# 最优 lambda
plot(cv_fit)
best_lambda <- cv_fit$lambda.min

# 非零系数（被选中的特征）
coef_list <- coef(cv_fit, s = best_lambda)
selected_features <- lapply(coef_list, function(c) {
  rownames(c)[which(c[, 1] != 0)]
})
```

Random Forest

Random Forest 不需要特征预筛选，能处理非线性关系，并提供特征重要性排序：

```
library(randomForest)

set.seed(42)
rf_model <- randomForest(X, y, ntree = 1000, importance = TRUE)

# OOB 错误率
print(rf_model)

# 特征重要性
imp <- importance(rf_model, type = 1)
top_imp <- head(sort(imp[, 1], decreasing = TRUE), 30)
barplot(top_imp, las = 2, main = "Top 30 features by importance",
        cex.names = 0.6)
```

交叉验证设计

多组学预测模型最容易犯的错误是信息泄露（data leakage）。特征选择必须在交叉验证的内层完成，不能用全部数据选完特征再做 CV。

```
library(caret)

# 正确做法：用 caret 的嵌套 CV
ctrl <- trainControl(method = "repeatedcv",
                     number = 5,
                     repeats = 10,
                     classProbs = TRUE,
                     summaryFunction = multiClassSummary)

# 在每个 fold 内部做特征选择 + 建模
set.seed(42)
rf_cv <- train(X, y,
               method = "rf",
               trControl = ctrl,
               tuneLength = 5,
               metric = "AUC")
print(rf_cv)
```

如果样本量很小（< 50），考虑 leave-one-out CV（LOOCV）或 repeated 5-fold CV 来减少方差。

多组学 vs 单组学的预测力比较

一个关键问题是：多组学整合是否真的比单组学预测更好？需要做对照实验：

```
# 分别用单层数据训练
rf_rna  <- train(t(rna_sel), y, method = "rf", trControl = ctrl, metric = "AUC")
rf_meth <- train(t(meth_sel), y, method = "rf", trControl = ctrl, metric = "AUC")
rf_prot <- train(t(prot_sel), y, method = "rf", trControl = ctrl, metric = "AUC")

# 比较
results <- resamples(list(
  MultiOmics = rf_cv,
  RNA_only   = rf_rna,
  Meth_only  = rf_meth,
  Prot_only  = rf_prot
))
summary(results)
dotplot(results, metric = "AUC")
```

如果多组学没有显著提升，说明信息冗余或者某一层已经足够。这本身也是有价值的结论。

生存预测

如果目标是生存时间而不是分类，用 Cox 回归配合 elastic net：

```
library(glmnet)

surv_y <- Surv(col_data$os_time, col_data$os_event)

cv_cox <- cv.glmnet(X, surv_y, family = "cox",
                    alpha = 0.5, nfolds = 5)
# C-index
max(cv_cox$cvm)
```

参考资源

- glmnet vignette
- caret 包文档
- randomForest 包
- 交叉验证中的信息泄露 (Hastie et al. ESL Ch.7)
- 多组学预测综述 (Picard et al. 2021)

09 整合结果的生物学解读与报告

跑完 MOFA、SNF 或机器学习模型之后，你手上有一堆因子、模块、聚类标签和特征列表。这些数字本身不是结论——需要把它们翻译成生物学语言，并以可复现的方式写进论文。

从因子到通路

MOFA 因子的权重列表本质上是一个 ranked gene list，可以直接送进 GSEA：

```
library(MOFA2)
library(fgsea)
library(msigdbr)

# 获取 Factor 1 在 mRNA 层的权重
weights <- get_weights(model, views = "mRNA", factors = 1, as.data.frame = TRUE)
gene_ranks <- setNames(weights$value, weights$feature)
gene_ranks <- sort(gene_ranks, decreasing = TRUE)

# Hallmark 基因集
hallmark <- msigdbr(species = "Homo sapiens", category = "H")
pathways <- split(hallmark$gene_symbol, hallmark$gs_name)

fgsea_res <- fgsea(pathways, gene_ranks, minSize = 15, maxSize = 500)
sig_pathways <- fgsea_res[padj < 0.05][order(NES, decreasing = TRUE)]
head(sig_pathways[, .(pathway, NES, padj)], 10)
```

如果 Factor 1 富集在 cell cycle 和 DNA repair 通路，而且它在 Mutations 层也有高方差贡献，你可以说"Factor 1 捕捉了与基因组不稳定性相关的跨组学变异"。

从聚类到亚型特征

SNF 或 consensus clustering 给出的聚类标签需要进一步表征：

```
# 每个亚型的差异特征
library(limma)

design <- model.matrix(~ 0 + factor(clusters))
colnames(design) <- paste0("C", 1:ncol(design))

fit <- lmFit(rna_final, design)
contrast_mat <- makeContrasts(C1 - C2, C1 - C3, C2 - C3, levels = design)
fit2 <- contrasts.fit(fit, contrast_mat)
fit2 <- eBayes(fit2)

# C1 vs C2 的 top 差异基因
topTable(fit2, coef = 1, number = 20)
```

把每个亚型的 marker 基因列出来，再做 GO/KEGG 富集，就能给亚型一个生物学标签（比如"免疫活跃型""代谢重编程型"）。

多层证据汇总

好的多组学分析不是把每层结果分别报告，而是展示"多层证据指向同一个结论"。一个常见的汇总方式：

```
# 某个基因在三层的一致性
gene <- "TP53"

# RNA 层：在亚型间差异表达
rna_p <- t.test(rna_final[gene, ] ~ clusters)$p.value

# 甲基化层：promoter 甲基化差异
meth_p <- t.test(meth_final[gene, ] ~ clusters)$p.value

# 蛋白层：蛋白丰度差异
prot_p <- t.test(prot_final[gene, ] ~ clusters)$p.value

cat(gene, "across layers:\n")
cat("  RNA p-value:", format(rna_p, digits = 3), "\n")
cat("  Methylation p-value:", format(meth_p, digits = 3), "\n")
cat("  Protein p-value:", format(prot_p, digits = 3), "\n")
```

写 Methods 段

多组学论文的 Methods 段需要覆盖以下内容。下面是一个模板：

```
Multi-omics integration was performed using MOFA2 (v1.8.0).
Input data included transcriptomics (N genes after filtering),
DNA methylation (N CpG sites, promoter-level aggregation),
and proteomics (N proteins). Each layer was independently
preprocessed: RNA-seq counts were variance-stabilized using
DESeq2; methylation beta values were Noob-normalized with minfi;
protein intensities were log2-transformed and batch-corrected
with limma::removeBatchEffect. Features were mapped to gene
symbols using biomaRt. The model was trained with K=15 factors,
slow convergence mode, and random seed 42. Downstream enrichment
analysis used fgsea with MSigDB Hallmark gene sets.
```

关键原则：

- 写清楚每层的预处理方法和工具版本
- 说明特征映射策略（probe → gene 用了什么注释）
- 报告整合方法的参数（因子数、迭代次数、收敛标准）
- 提供代码仓库链接

可视化建议

多组学论文常见的图：

图类型	用途	工具
方差解释热图	展示因子在各层的贡献	MOFA2 内置
多层 heatmap	同一批样本在不同组学的模式	ComplexHeatmap
Sankey / alluvial 图	展示亚型在不同方法间的对应	ggalluvial
网络图	融合网络结构	igraph、Cytoscape
Forest plot	多组学 Cox 回归系数	forestplot

```
library(ComplexHeatmap)

# 多层 heatmap：同一批样本，上面是 RNA，中间是甲基化，下面是蛋白
ht1 <- Heatmap(rna_final[top_genes, order(clusters)], name = "RNA",
               show_column_names = FALSE, cluster_columns = FALSE)
ht2 <- Heatmap(meth_final[top_genes, order(clusters)], name = "Meth",
               show_column_names = FALSE, cluster_columns = FALSE)
ht3 <- Heatmap(prot_final[top_genes, order(clusters)], name = "Prot",
               show_column_names = FALSE, cluster_columns = FALSE)

ht_list <- ht1 %v% ht2 %v% ht3
draw(ht_list, column_title = "Multi-omics heatmap by subtype")
```

可复现性清单

发表前检查：

- 原始数据是否上传到公共数据库（GEO、PRIDE、GDC）
- 分析代码是否在 GitHub/Zenodo 上公开
- R session info 是否记录（`sessionInfo()`）
- 随机种子是否固定
- 中间结果（模型文件、聚类标签）是否保存

参考资源

- fgsea Bioconductor
- msigdbr 基因集
- ComplexHeatmap 完整手册
- ggalluvial 包
- 多组学论文写作指南 (Subramanian et al. 2020)